TV3P: An Adaptive Assistant for Personalized TV

Zhiwen Yu, Student Member, IEEE and Xingshe Zhou, Member, IEEE

Abstract[1] — *The abundance of DTV (Digital Television) programs precipitates a need for new tools to help people personalize interesting TV content. We developed an adaptive assistant: TV3P (TV Program Personalization for PDR), which observes users' viewing behaviors in the background, updates users' profiles continuously and autonomously, and then filters and recommends programs for different users according to their respective preference information. The novel aspect of this system is the evaluation of how much time and effort it takes the system to learn new preferences once it already is biased by old preferences. This has not been proposed in any other recommender systems before. It was also proved to match real world users whose preferences can change over time. Another attractive aspect of TV3P is its employing an implicit and explicit profiling scheme.*

Index Terms — Personalization, TV3P, User profiling, Vector space model.

I. INTRODUCTION

Digital television (DTV) and the rapid growth of communication technologies, especially the merge of DVB-C and DVB-S, have created an overabundance of programs and information available from which each consumer can choose. This precipitates a need for tools to help people find interesting programs.

Historically, television viewers identified television programs of interest through two ways: browsing printed television program guides or channel surfing. The printed television program guides provide limited information about programs, such as the programs' broadcast time and date, channel and title. Further more, for 200 channels within today's digital television, the printed TV guide is a thick book of 140 pages, which take the user 33 minutes to browse [1]. Also, for 200 channels, channel surfing (switching up and down until the viewer finds something to watch) may take a long time. Most users have no patience to use above two ways.

More recently, electronic program guides (EPGs) have become available. While EPGs allow viewers to identify desirable programs more efficiently than conventional printed guides, they still lack of intelligence. The TV viewer still has to look for interesting programs manually.

Intelligent personalization techniques can greatly improve the efficacy of retrieving and searching interesting programs. We designed and implemented ***TV3P*** (TV Program Personalization for PDR, Personal Digital Recorder), which can provide users with adaptive and personalized TV viewing assistance. TV3P observes users' viewing behaviors in the background, updates users' profiles continuously and autonomously, and then filters and recommends programs for different users according to their respective preference information.

The rest of this paper is organized as follows. Section 2 discusses previous related work with this paper. Section 3 provides the architecture of the system. Section 4 describes the filtering and recommendation strategy using VSM. The user profiling scheme by integration of explicit input/modification, explicit feedback, and implicit feedback is presented in Section 5. The prototype implementation and evaluation results obtained are provided in Section 6. Finally, Section 7 points out directions for future work.

II. RELATED WORK

Much work has been done in the area of information personalization and they have been applied to a wide range of fields, such as e-mail, usenet news, XML documents, E-Commerce, and web. Besides above fields, TV is another information source needing personalization. Several personalized TV systems have been built in recent years to help users deal with the overabundant TV programs.

Philips multi-agent TV recommender system that encapsulates three user information streams—(implicit viewing history, explicit preferences, and feedback information on specific shows) into adaptive agents and builds a framework that allows for these multiple agents to collaborate and generate a combined program recommendation for a TV viewer [2]. Our system is based on a different multi-agent model. The learning algorithm we adopted, based on relevance feedback method, is also different.

TV-Advisor [3] makes use of explicit techniques to generate recommendations for a TV viewer. Such techniques require the user to take the initiative and explicitly specify their interests, in order to get high quality recommendations. In our system, we adopt implicit learning for user profile. Implicit learning lessens the burden on the user and tries to infer the user's preferences from a viewer's TV viewing history.

PTV [4] uses a content-based plus collaborative filtering approach to generate TV show recommendations. Though they seek similar user profile information like what we do in our system, they do not include a 'dynamic', learning algorithm that tracks a person's changing TV preferences over time.

In their paper [5], L. Ardissono *et al*. present a multi-agent architecture of a system for the generation of adaptive EPGs. The user profile is about the user's preferences at the time of day s/he wants to watch TV. The authors put the focus on the architecture and representation TV events using DVB other than algorithms.

[1] Z. Yu is with the College of Computer of Northwestern Polytechnical University, Xi'an, Shaanxi, P.R.China. (e-mail: yuzhiwen77@yahoo.com.cn).

X. Zhou is with the College of Computer of Northwestern Polytechnical University, Xi'an, Shaanxi, P.R.China. (e-mail: zhouxs@nwpu.edu.cn).

Contributed Paper
Original manuscript received November 16, 2003
Revised manuscript received December 8, 2003

TV Scout [6] is a recommendation system providing users with personalized TV schedules. The highlight of TV Scout is its solution for "cold-start" problem of information filtering systems.

In comparison with above systems, the strongest aspect of our system is the evaluation of how much time and effort it takes the system to learn new preferences once it already is biased by old preferences. This has not been proposed in any other recommender systems before. It was also proved to match real world users whose preferences can change over time.

III. TV3P ARCHITECTURE

Fig. 1 illustrates TV3P's schematic and high-level architecture. There are four agents in the whole multi-agent system, namely, filtering agent, recommendation agent, interface agent, and profiling agent. The filtering agent filters the incoming live TV programs broadcast to PDR, only recording the programs that the agent thinks the user would like. The recommendation agent is used to generalize recommendation list from the stored local programs to the user. The interface agent deals with the GUI interaction with the user, for example, it provides the GUI interface for the users to input or modify their profiles. The profiling agent is responsible to update the user's profile according to his/her viewing behaviors, so as to adapt to the user's most recent preferences. In program filtering and recommending process, the user profile database provides user preference information for matching and evaluating programs.

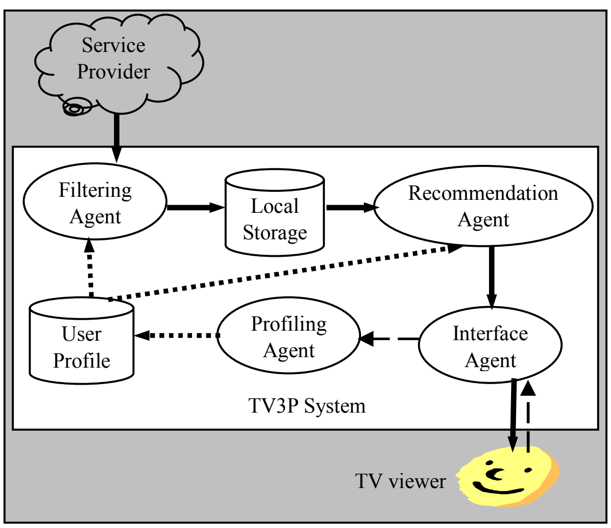

Fig. 1. TV3P architecture

IV. FILTERING AND RECOMMENDATION STRATEGY

For the purpose of interoperability, in TV3P, the program description (program metadata) and user profile are both represented in XML, and complied with TV-Anytime specifications, which uses the MPEG-7 Description Definition Language (DDL) to describe the metadata structure as well as the XML encoding of metadata [7]. The program metadata in our system includes the following fields (elements): *Title*, *Genre*, *Actor*, *Keyword*, and *Duration*. *Duration* indicates the time, in minutes, of the program will last, which is used for user profiling in Section 5. The user profile contains a set of terms; in the term field, each term has *weight* as its attribute. The terms are sorted by weight in descending order in the user profile.

We adopt the Vector Space Model (VSM)[8] as the feature extraction and object information presentation method. In the VSM paradigm, we identify an object by a set of terms. Weights are assigned to terms as importance indications.

In the user profile, there may be a lot of terms, which indicate the user's interests. The terms have weights respectively, in other words, each term is defined as a 2-tuple (term, weight). So the user profile can be represented as a vector of these 2-tuples, if there are m distinct terms in the profile, then it will be represented as a vector:

$$P = ((t_1, w_1), (t_2, w_2), \ldots, (t_m, w_m)) \quad w_i \geq w_{i+1} \qquad (1 \leq i \leq m) \tag{1}$$

where t_i is a term, w_i is the weight of term t_i. The weights describe the relative importance of the terms in the profile.

For computational reasons, we can take the top n highest weighted terms to represent the user's preference. So the user's profile can be conceptually represented as the following vector:

$$P = (w_1, \ldots, w_n) \tag{2}$$

where w_i is the weight of term t_i in the profile.

Similarly, a program can be also represented as a vector with n items, which are the same as those in the profile vector, that is term t_i in the profile vector and content vector is the same:

$$C = (u_1, \ldots, u_n) \tag{3}$$

where u_i is the weight assigned to term t_i. Since terms are not all equally important for program representation, for instance, terms in *Actor* field may be more important than those in the *Keyword* field, we should assign important factors to the terms in different fields to reflect their relative importance for program identification. We define a field set: S={Title, Genre, Actor, Keyword}, and an importance factor set: $W=\{W_x \mid x \in S\}$. W_x is the importance factor assigned to terms in *X* field to reflect their relative importance. W_x ranks as follows: $W_{Title} > W_{Genre} > W_{Actor} > W_{Keyword}$.

With above definitions, the weight u_i is assigned complying with the following rules:

- Rule (i): if term t_i is merely included in *x* field of the program's metadata, then $u_i = W_x$,
- Rule (ii): if t_i is included in two or more fields, then $u_i = Max\{W_x\}$ (the maximum value of W_x that t_i is included in corresponding fields),
- Rule (iii): if t_i is not included in any field, then u_i=0.

In the classical vector space representation, the similarity between the two vectors of program and profile indicates the degree of relevance between the program and the profile. A commonly used similarity metric is the cosine of the angle between the two vectors. Given a program $C=(u_1,...,u_n)$ and a profile $P=(w_1,...,w_n)$, the cosine similarity can be calculated as follows:

$$Sim(C,P)=\frac{C\times P}{\|C\|\times\|P\|}=\frac{\sum_{i=1}^{n}u_i w_i}{\sqrt{\sum_{i-1}^{n}u_i^2\sum_{i=1}^{n}w_i^2}} \quad (4)$$

When a program arrives, if the calculated similarity is above the preset threshold θ (such as 0.40), we consider that the program is relevant to the user's profile; in other words, the user is likely interested in the program. Then the filtering agent will record the program for the user. In the same way, the recommendation agent can evaluate the available programs in the local storage using (4), and then suggest the l highest similarity programs to the user.

Fig. 2 illustrates the process of feature extraction and similarity measurement (Supposing W_{Title}=1.25, W_{Genre}=1, W_{Actor}=0.75, $W_{Keyword}$=0.5).

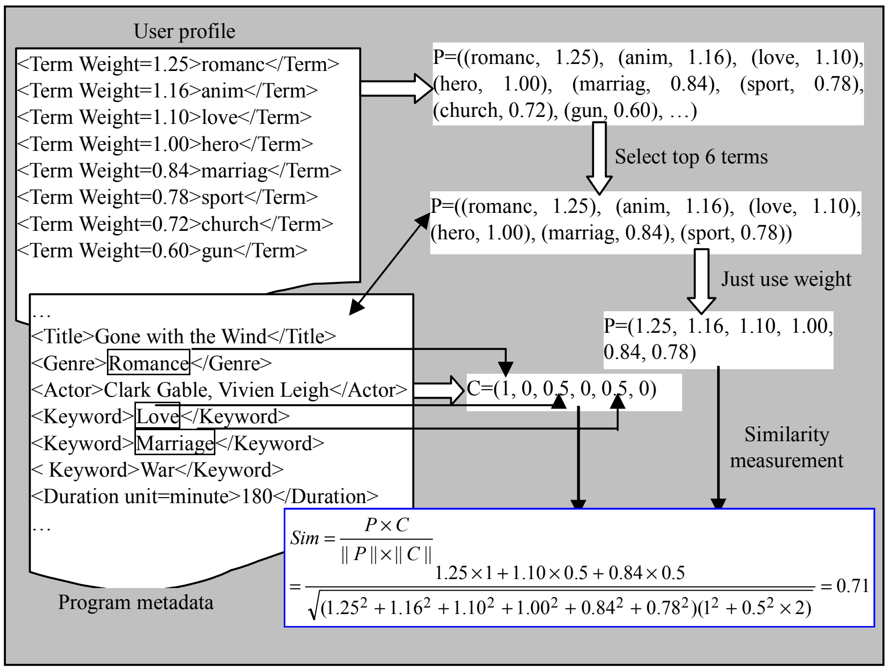

Fig. 2. How the feature extraction and similarity be done

V. USER PROFILING

Three knowledge sources can be used to update user profile: explicit input/modification, explicit feedback, and implicit feedback. The user profiling by integration of all these three modes is shown in Fig. 3.

Profile update through explicit input/modification (input interests when registration or modify user preference after log in through GUI) can be done easily and directly. While the preference learning through explicit feedback and implicit feedback is more complicated. In TV3P, the profiling agent utilizes relevance feedback [9] to learn user's preferences according to explicit feedback and implicit feedback. User profile update is done through the modification of the preference terms and their weights respectively. After term-weight modification, the terms in the user profile will be re-ordered (also in descending order) according to their current weights.

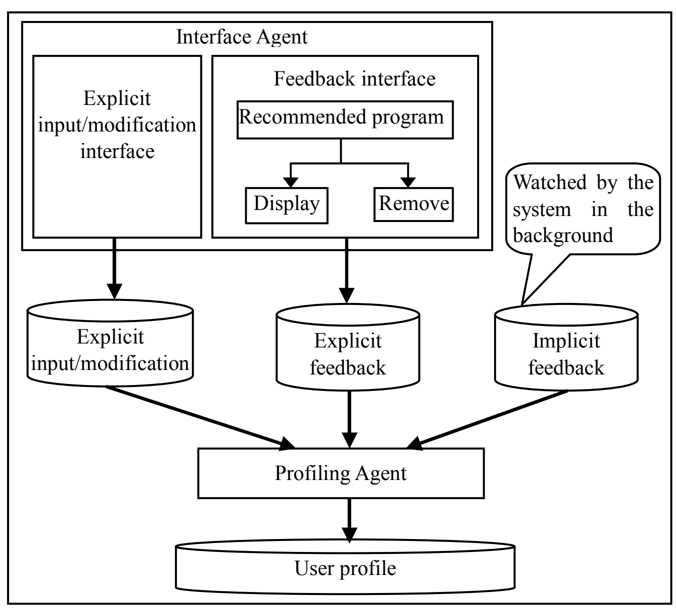

Fig. 3. Profiling by integration of explicit input/modification, explicit feedback, and implicit feedback.

A. Implicit profiling

In our system, when the user has watched a program for a period of time, and switched to another one, then the user profile will be refined and revised based on the implicit feedback automatically observed by the system in the background. The algorithm is depicted as follows: for those terms in the metadata of the program just recommended,

(1) If a term already exists in the profile, its weight is modified in proportion to the feedback.

$$w_i' = (1-\alpha)\times w_i + \alpha\times\Delta w_i \tag{5}$$

$$\Delta w_i = \beta\times f(i) \tag{6}$$

$$\beta = \frac{T_r}{T_t} \in [0,1] \tag{7}$$

$$f(i) = \frac{I_{\max} - i}{I_{\max}} \qquad 1 \le i \le I_{\max} \tag{8}$$

where w_i' is the weight of term t_i after update, and w_i is the weight of term t_i before update. α ($0 \le \alpha \le 1$) is the learning rate that determines how quickly the user profile forgets old preferences and tracks new ones. β is the ratio of user's real watching time (T_r) to the program's total duration time (T_t). β can be considered as the user's evaluation to the program which he/she has viewed. $f(i)$ reflects the influence of the order of the term in user's viewing history to the weight update process. The more important terms with larger weight reasonably have more influence on the profile learning. So $f(i)$ should decrease with increasing order of the term. In the expression of $f(i)$, i is the order of term t_i in the user's profile; $I_{\max}$ is the maximum of i, in other words, it is the total number of terms in the user's profile.

(2) If a term does not exist in the profile,

(i) Firstly, identify the individual words occurring in the program metadata. Words that belong to the *stop list*, which is a list of high-frequency words with low content discriminating power, like "the", are deleted. Then use the stemming routine to reduce each remaining word to word-stem form, that is, the remaining words are reduced to their stem by removing prefixes and suffixes. This is used for decreasing redundancy. We use the stopping and stemming algorithm implemented in [10].

(ii) Secondly, calculate the term's weight

$$w_i = \alpha\times\beta\times f(i) \tag{9}$$

here w_i is the weight of the new term t_i, α and β have the same meanings as step (1). Since term t_i is not in the profile before, so $f(i)$ can't be calculated as above, we define it as a default value ε.

(iii) Thirdly, if the calculated w_i is higher than a preset threshold λ, we will add it to the user's profile, otherwise discard it, because it is too trivial.

B. Explicit profiling

The profiling agent can also updates terms and their weights after explicit user actions. The update rule is the same as equation (5-1). When a program is recommended to the user according to his/her profile, two buttons for selection are also provided, which are *Display* and *Remove* buttons. Now, three types of user possible actions will take place: (i) Display: clicking on the *Display* button means to show the program; (ii) Let it be: doing nothing means to leave the program to display automatically; (iii) Remove: clicking on the *Remove* button means to delete the program. Each action has its own effect on Δw_i for all of that program's keywords:

- Display: Clicking on the Display option is strong positive feedback: set $\Delta w_i = +2$.
- Let it be: The user leave the program to display automatically, it is weak positive feedback: set $\Delta w_i = 1$.
- Remove: The user actively deletes the program from recommendation list, it is strong negative feedback: set $\Delta w_i = -2$.

C. Implicit + Explicit profiling

In order to learn user's preference by considering both implicit feedback and explicit feedback, we define an implicit + explicit profiling algorithm:

$$w_i' = (1-\alpha)\times w_i + \alpha\times(W\_I\times\Delta w_i\_I + W\_E\times\Delta w_i\_E) \tag{10}$$

$$W\_I + W\_E = 1 \qquad (0 \le W\_I \le 1, 0 \le W\_E \le 1) \tag{11}$$

The combinative profiling integrates implicit profiling and explicit profiling. In above equations, Δw_i_I is computed from implicit feedback, while Δw_i_E is from explicit feedback. W_I and W_E are weighting factors reflecting the relative importance of implicit profiling and explicit profiling.

Fig. 4 illustrates the combinative profile update algorithm (α=0.25, W_I=0.4, W_E=0.6). In the case in Figure 4, two programs are recommended to the user: "Gone with the Wind" and "Animal World". The user first clicks to remove the latter program, and then clicks to display the former one. From these actions the profiling agent infers a strong dislike for term "Animal", and a strong interest for terms "Romance", "Love", and "Marry". For legibility and simplicity, this figure denotes only eight terms in the user profile, and just illustrates the weight value changing course of term "romanc" and "anim".

VI. IMPLEMENTATION AND EVALUATION

A. Prototype Implementation

Since DTV services are just starting, we have simulated a program source to demonstrate our system. The prototype environment involves 1 PC, 1 TriMedia card, and 1 TV display. The PC is equipped with Intel Pentium III 800 processor, 128Mb RAM, working as the host (that is PDR). The TriMedia card has 8Mb RAM, analogue audio, video input and output ports and a TM1000 processor, for video decoding. The TV display is used for displaying the content.

The Operating System we used is Mandrake Linux release 8.2 for i586, kernel 2.4.18-6mdk.

The program source simulator plays the roll of a future TV-Anytime service provider: it continuously broadcasts program to the PDR device. Our system filters the broadcast content and learns the user's preference. If the live broadcast content has a higher preference value, it will be recorded for the user. The program is sent to be displayed on the TV set via the TriMedia card of our PC/Linux prototype host.

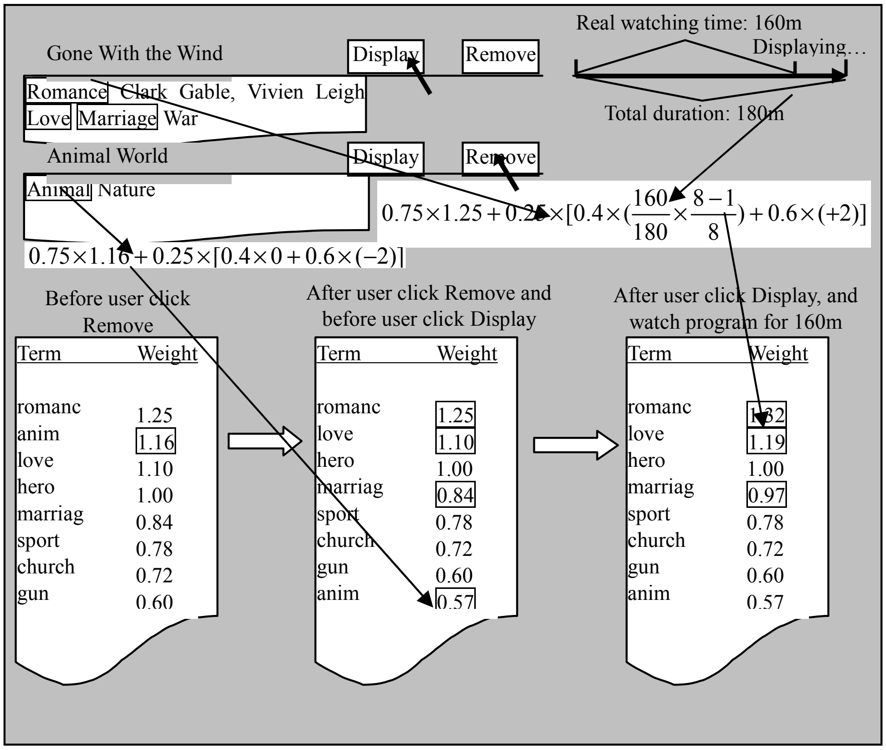

Fig. 4. How the profiling agent changes a profile according to user implicit feedback and explicit feedback.

Fig. 5a is the registration dialog. It allows a user to register a user account through which he/she can make use of the system. It also provides the GUI interface for the users to input their initial interests. If user is interested in some keyword that it does not appear in existed list, user can add it.

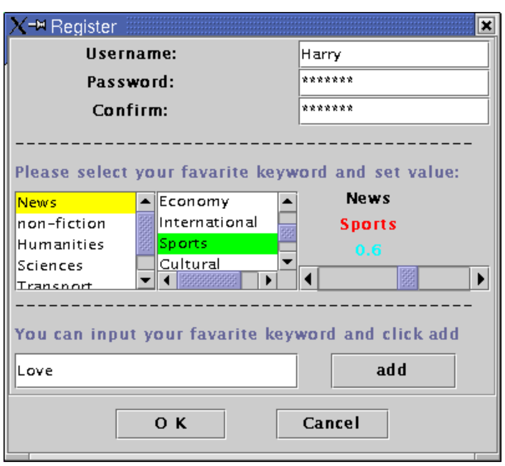

Fig. 5a. User registration

Fig. 5b shows the system status of the prototype system. The information includes who has logged in, what is the current program being broadcast, what is the current program being displayed, which program has been skipped and which has been recorded. In this dialog, a chart is used to show similarity and title of recent 6 simulated broadcast programs (when the mouse pointer is on the number along abscissa, the title of the program will appear); dots above the horizontal line means the respective programs are recorded or recommended directly (because the simulated broadcast program's similarity is higher than the min, here min=0.4), dots below the horizontal line means the respective programs are discarded (the similarity is lower than the min).

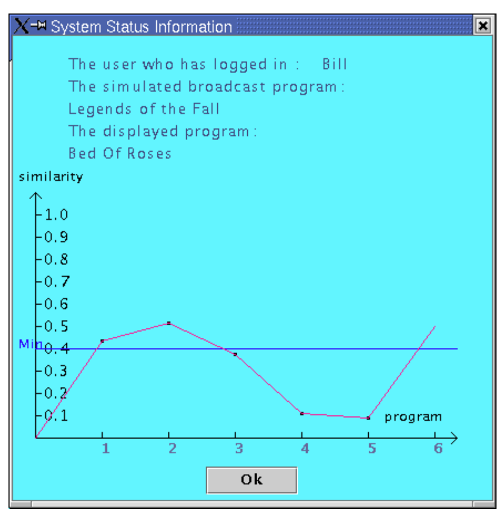

Fig. 5b. System status

When a user logs in, TV3P shows the profile it learned (see Fig. 5c). The user name and all keywords in user profile are displayed; Left column places the user's interesting keywords, while right column places the importance weight values respectively, and the keywords are listed by weight descending.

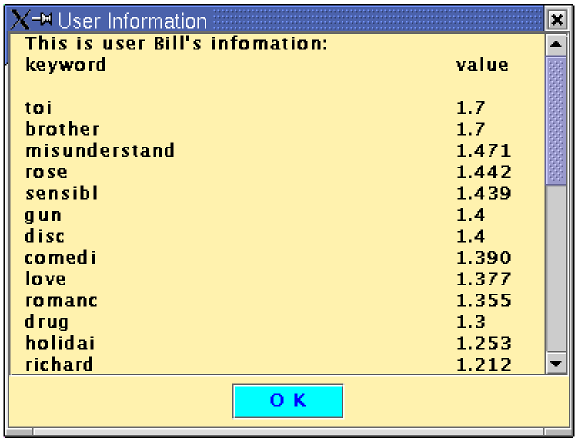

Fig. 5c. Display user profile

B. Performance Evaluation

From February 14 to July 31 2003, 20 real users (students in the author's lab, 16 males and 4 females) were asked to watch content of their choices with the system. Since 5 of them spent little time using the system and little profile was obtained, we mainly depended on the other 15 users' testing data. The entire experiment involved a total of 1682 distinct film segments lasting from 20 seconds to 5 minutes.

From user's point of view, filtering effectiveness is very important. It indicates the extent of personalization. There are two criterions for evaluating filtering effectiveness, which are precision and recall [11]. In general, precision can be used as a measure of the ability of our system to present only relevant programs. Recall can be used as a measure of the ability of our system to present all relevant programs.

$$\text{Precision} = \frac{\textit{number of relevant programs recorded}}{\textit{total number of programs recorded}}$$

$$\text{Recall} = \frac{\textit{number of relevant programs recorded}}{\textit{number of relevant programs in collection}}$$

Since the two measures are often conflicting, we use Recall-Precision Graph, which integrates both precision and recall, to evaluate filtering effectiveness. In the graph, each dot is a pair of recall-precision value. The plots of different runs can be superimposed on the same graph to determine which run is superior. Curves closest to the upper right-hand corner of the graph (where recall and precision are maximized) indicate the best performance.

In the first case, the users' current desires are consistent with foregoing profile. Fig. 6a shows the results of the experiments. The figure consists of three graphs. Each graph is a plot of precision versus recall. Comparing the results for the three learning algorithms, we can see that the implicit + explicit profiling is superior to the other two single feedback profiling. This can be seen from that the implicit + explicit profiling's curve is closer to the upper right-hand corner of the graph.

For the second case, before these experiments, the users have used the system for some time and were fond of movies in Class A. Now, they are asked to shift their preference to Class B. They continued to use the system, which has been specialized to Class A, expecting the system to specialize to Class B. This experiment would evaluate the ability of the system to specialize when the initial profile is not empty, but has a previous bias. The results are shown in Fig. 6b. The figure also consists of three graphs. At first, the explicit profiling has better performance than the other two. But with time going, the performance of implicit + explicit profiling increases. In the last sessions (where recall value is 0.32, 0.73, 0.91), the implicit + explicit profiling has the best performance.

Through the experimental results, we concluded that our system, especially employing an implicit and explicit profiling scheme, could keep track of user preference changing over time, and perform good filtering effectiveness.

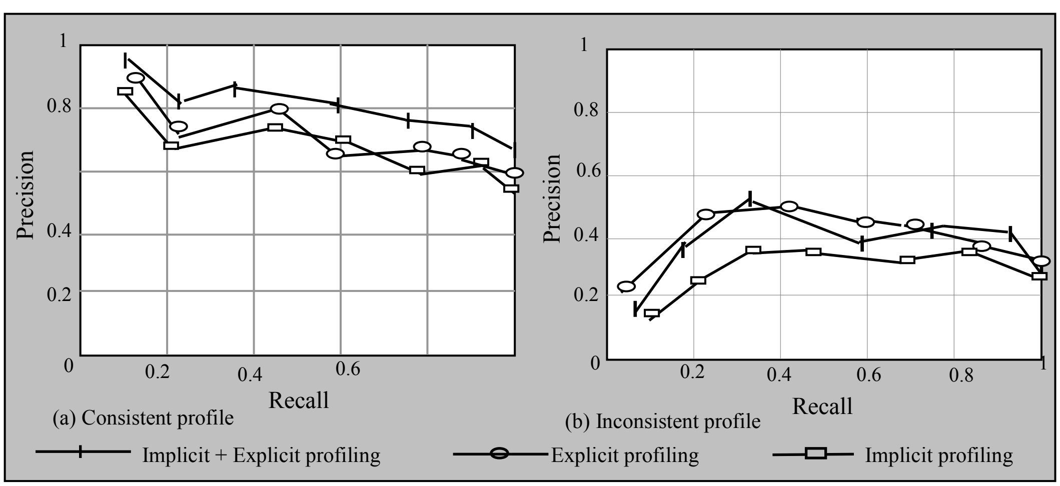

Fig. 6. Recall-precision graph. (a) The user's current desires are consistent with previous profile. (b) The user's current desires are inconsistent with previous profile.

VII. FUTURE WORK

We plan to implement a few improvements to TV3P. First, in the system, multiple users are allowed to log in the system as different users with user ID and password. The efficient and proper merging scheme and method of those different user profiles are being considered, which will produce a profile that may reflect the common interests for a group of users, for example, a family or all the students in one dormitory, etc. Second, we plan to improve the trustworthiness of the system. Consumers of TV personalization system will demand certain characteristics, including reliability, privacy, and ease of use (usability). These features can be encapsulated in the term "trustworthiness".

REFERENCES

[1] Jiangshan Xu, Liangjie Zhang, Haiming Lu and Yanda Li, "The Development and Prospect of Personalized TV Program Recommendation Systems", *In Proceedings of the IEEE 4th International Symposium on Multimedia Software Engineering* (MSE'02), December 11-13, 2002, Newport Beach, California, USA.

[2] K. Kurapati, S. Gutta, D. Schaffer, J. Martino and J. Zimmerman. "A Multi-Agent TV Recommender". *In Proceedings of the User Modeling 2001: Personalization in Future TV Workshop*, Sonthofen, Germany, 2001.

[3] Das, D. and ter Horst, H. "Recommender Systems for TV". *In Proceedings of AAAI*, 1998.

[4] Cotter, P., and Smyth, B., "PTV: Personalised TV Guides". *In Proceedings of the 12th Conference on Innovative Applications of Artificial Intelligence, IAAI 2000*, Austin, Texas, 2000.

[5] L. Ardissono, F. Portis and P. Torasso, "Architecture of a System for the generation of personalized Electronic Program Guides". *In Proceedings of the User Modeling 2001: Personalization in Future TV Workshop*, Sonthofen, Germany, 2001.

[6] P. Baudisch and L. Bruekner, "TV Scout: Guiding Users from Printed TV Program Guides to Personalized TV Recommendation". *In Proceedings of the AH'2002 Workshop on Personalization in Future TV*, Málaga, Spain, May 2002

[7] TV-Anytime Metadata Specifications Document, SP003v12 PartA, TV-Anytime Forum, August 2001.

[8] G Salton, "Automatic Text Processing: The transformation, analysis, and retrieval of information by computer", Addison-Wesley, Massachusetts, USA, 1989

[9] J.J.Rocchio. "Relevance feedback in information retrieval". *In The Smart System--Experiments in Automatic Document Processing*. Prentice Hall, 1971.

[10] William B. Frakes and Ricardo Baeza-Yates, "Information Retrieval: Data Structures and Algorithms", Prentice-Hall, 1992.

[11] C.J. van Rijsbergen. Information Retrieval. *Butterworths*, second edition, 1979.

[12] Filippo Menczer, Alvaro E. Monge and W. Nick Street, "Adaptive Assistants for Customized E-Shopping", *IEEE Intelligent Systems*, 2002 (6): 12-19.

[13] Zhou Xingshe, Yu Zhiwen, Gu Jianhua, Wu Xiaojun, Zhang Yong. "A Multi-Agent System for Personalized and Private Service in PDR". *In Proceedings of the 4th IEEE International Conference on Information Technology: Computers and Communications* (ITCC2003), PP635-639, IEEE Computer Society Press, April 28-30, 2003, Las Vegas, Nevada, USA.

Zhiwen Yu is a PhD candidate at the College of Computer, Northwestern Polytechnical University, China. His research interests include personalized TV, intelligent information technology, adaptive intelligent agents, personalization technology, and embedded computing. He received his B.S. and M.S. degrees in computer science and technology from Northwestern Polytechnical University, in 2000 and 2003, respectively. He is a student member of IEEE.

Xingshe Zhou is a professor at the College of Computer, Northwestern Polytechnical University, China. His research interests include adaptive intelligent agents, distributed computing, and embedded computing. He received his M.S. degree in computer science and technology from Northwestern Polytechnical University. He is a member of IEEE.